XXXVIII Jornadas de Automática
https://doi.org/10.17979/spudc.9788497497749.0130

Implementación y evaluación de controladores basados en eventos en la norma IEC-61499

Oscar Miguel-Escrig, Julio-Ariel Romero-Pérez, Esteban Querol-Dolz
Departamento de Ingeneiría de Sistema Industriales y Diseño
Universidad Jaume I
omiguel@uji.es, romeroj@uji.es, estebanqueroldolz@gmail.com

Resumen

En este artículo se presenta un estudio sobre el desempeño de controladores PID basados en eventos implementados según el estándar de programación IEC-61499. En el trabajo se abordan los detalles de la implementación y se presentan resultados experimentales que demuestran las ventajas de este tipo de algoritmos respecto a los controladores basados en tiempo, fundamentalmente en cuanto a la reducción del coste computacional sin afectar de forma significativa al comportamiento del lazo de control.

Palabras clave: PID, control basado en eventos, tiempo real, IEC-61499

1. INTRODUCCIÓN

En la última década se han realizado numerosa investigaciones sobre el control basados en eventos (CBE) de sistemas continuos. A diferencia de los controladores basados en tiempo usados tradicionalmente, en los que la ejecución del algoritmo de control se realiza a un periodo constante, en los controladores basados en eventos el algoritmo de control se ejecuta sólo tras la ocurrencia de eventos asíncronos que indican cambios significativos en el estado del sistema; un ejemplo típico es el cruce de niveles por parte de la señal de error (diferencia entre la referencia o set-point y la salida controlada), [3].

A la par de los avances teóricos en el CBE, durante los últimos años se ha estado desarrollando un nuevo estándar para la programación de sistemas de automatización y control distribuidos, conocido como IEC-61499 [9]. Dicho estándar introduce conceptos novedosos respecto a su predecesor, el IEC-61131 [6], el cual es ampliamente usado en la actualidad en la programación de Autómatas Programables; como por ejemplo la forma de ejecución, mientras que el antiguo está basado en ciclos de SCAN y el nuevo basado en el uso y manejo de eventos, y que además contiene características que favorecen su uso en el diseño de aplicaciones distribuidas y reconfigurables.

En este trabajo se realiza un estudio sobre la implementación y el desempeño de algoritmos de CBE usando el estándar IEC-61499. Esto permitirá introducir en el IEC-61499 estrategias de control de sistemas continuos más acorde con las características del propio estándar (control de ejecución por eventos), en sustitución de los controladores basados en tiempo (con ejecución periódica) que se han venido usando hasta ahora. En el estudio sobre el desempeño de los controladores se tendrán en cuenta tanto los aspectos de coste computacional como del comportamiento del sistema de control.

Dado que los controladores PID son los más utilizados en aplicaciones industriales, y que actualmente existen varios trabajos en los que se proponen distintas versiones de PID basados en eventos, hemos considerado conveniente realizar el estudio sobre la implementación y el comportamiento de este tipo de controladores bajo el IEC-61499.

2. BREVE DESCRIPCIÓN DE LA NORMA IEC-61499

En el centro de la norma se encuentra el modelo de bloques de funciones (Function Block Model - FBM), donde un bloque de funciones (Function Block - FB) es una unidad funcional de software, con su propia estructura de datos que puede ser manipulada por uno o más algoritmos que dotan al FB de su funcionalidad. Un FB define un *tipo* a partir del cual se pueden crear una o varias instancias del FB.

En los FBs confluyen dos tipos de parámetros, los datos y los eventos, los cuales regulan la ejecución del bloque. Un FB puede tanto recibir como enviar estos tipos de parámetros y asociarlos de manera que la recepción o envío de un evento se corresponda con el refresco de los datos de entrada o salida asociados. Los FBs encapsulan una funcionalidad concreta, lo que puede dar lugar a que estos FBs contengan variables internas, cuyo tipo y tratamiento vienen regulados por las características del FB. Como en la norma IEC-61499 no se permite el uso de variables globales para la aplicación, los algoritmos que definen la funcionalidad del FB, sólo tendrán acceso a las entradas,

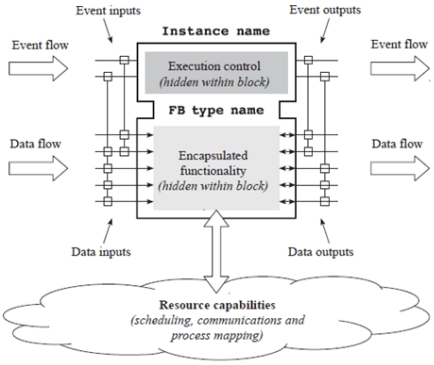


Figura 1: Estructura de un FB. [9]

las salidas y a las variables internas, siendo todos ellos datos retenidos entre llamadas al FB.

En la figura 1 se muestras los elementos anteriormente descritos de un FB. En la parte superior se representa el *control de ejecución* (Execution control) que gestiona la entrada y salida de eventos. En la parte inferior se representa la *funcionalidad* del FB, formada por los algoritmos encargados de procesar los datos de entrada. Cuando ocurre un evento de entrada, se actualizan los datos de entrada asociados al evento y se ejecutan los algoritmos vinculados al mismo. La asociación de eventos y datos se representa mediante las lineas verticales que aparecen en la entrada y salida del FB y que unen eventos y datos mediante pequeños cuadrados. Como resultados de la ejecución de los algoritmos se generan los eventos y datos de salida.

Los BFB se caracterizan por tener una estructura y comportamiento que están definidas como respuesta a la recepción de eventos entrantes. Internamente, esto se materializa en un gráfico de control de ejecución (Execution Control Chart - ECC), cuyas transiciones pueden estar reguladas por la recepción de dichos eventos o por las variables que se traten durante su ejecución.

Los FBs se agrupan en un modelo que configura una aplicación de la norma IEC-61499 (Application Model - AM). Una aplicación está definida por las diferentes instancias de FB que la componen (pudiendo haber múltiples instancias de un mismo tipo de bloques) y por la interconexión de los mismos. Como la funcionalidad total del programa queda definida completamente en función de los FBs, no se necesita de variables globales o locales fuera de los mismos.

3. CONTROLADORES PID BASADOS EN EVENTOS

Una de las primeras contribuciones al desarrollo de los controladores PID basados en eventos fue introducida por Årzén en [1] como una forma de reducir el uso de la CPU de los sistemas de control basados en computador sin afectar de forma significativa el comportamiento del bucle de control. En su artículo, Årzén puso de relieve algunas de las consideraciones más importantes que se debía tener en cuenta en los controladores PID basados en eventos. Entre éstas se pueden destacar los errores que se producen en los cálculos de los términos integral y derivativo cuando el tiempo entre muestras se incrementa. Varios trabajos posteriores estuvieron dirigidos a resolver los problemas revelados por Årzén, fundamentalmente el relacionado con el error en el cálculo de error del término integral. Deben ser destacados en este sentido los trabajos publicados por Durand [4, 5] y Vasyutynskyy [7, 8].

En concreto, el algoritmo de Årzen plantea una llamada periódica al controlador no siéndolo así la ejecución del cálculo. Årzen propone una lógica de detección de eventos basada, por una parte, en que la diferencia de errores entre la última ejecución y la llamada actual supere un umbral y por otra parte impone una condición basada en el tiempo máximo que el controlador puede estar sin ejecutarse.

Además, y a diferencia del controlador PID basado en ciclos de SCAN, el cual pre-calcula algunos coeficientes, en el algoritmo de Årzen se deben recalcular cada vez ya que dependen del tiempo entre ejecuciones. Por todo ello el código resultante queda de la forma siguiente:

```
e = ysp - y;
% calcula acción de control
hact = hact + hnom;
if abs(e - e_old) > elim || hact > hmax
   %coeficientes del pid discreto
   ad=Td/(N*hact+Td);
   bd=K*Td*N/(Td+N*hact);
   bi=K*hact/Ti;
   %algoritmo de control PID discreto
   up= K*(b*ysp - y);
   ud=ad*ud + bd*(c*(ysp-ysp_old)-(y-y_old));
   ui=ui+bi*(ysp - y);
   u=up + ui + ud;
   if (u<u_sat_min)
      u=u_sat_min;
      if (ysp-y<0)
         ui=ui-bi*(ysp - y);
      end
   end
   if (u>u_sat_max)
      u=u_sat_max;
      if (ysp-y>0)
         ui=ui-bi*(ysp - y);
      end
   end
   y_old=y;
   ysp_old=ysp;
   hact = 0;
   event=1;
```

```
end
u_out = u;
```

4. IMPLEMENTACIÓN DE CBE EN LA NORMA IEC-61499

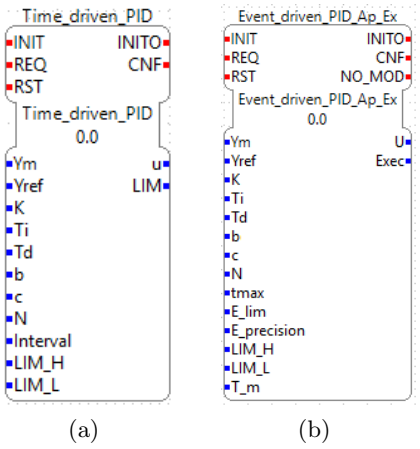
El desarrollo de controladores con la norma IEC-61499 se lleva a cabo mediante la implementación de nuevos bloques de funciones, ya que estos constituyen la unidad básica de programación. En este estudio, vamos a implementar y comparar un controlador basado en un tiempo clásico, con período constante, con el controlador basado en eventos de Årzen, desarrollado en [1].

Para la implementación y evaluación de comportamiento de los controladores hemos utilizado el sotfware 4DIAC, un entorno de desarrollo *open-source*, basado en el entorno de desarrollo Eclipse. Una alternativa de software más enfocada al ámbito industrial sería la utlización del software NxtOne, el cual también es capaz de desarrollar aplicaciones en la norma IEC-61499 y además soporta hardware de diferentes fabricantes industriales como BECKHOFF, WAGO o SIEMENS.

4DIAC permite la creación de bloques, sistemas, aplicaciones y el resto de modelos. Las aplicaciones desarrolladas en 4DIAC son interpretadas por el runtime FORTE que debe estar ejecutándose en el dispositivo de control. Este runtime pude descargarse de la página del proyecto 4DIAC.

La Figura 2 muestra la estructura externa de dos controladores. A la izquierda tenemos el controlador periódico y a la derecha el controlador de Årzen. Al tratarse en este caso de BFB (Basic Function Block), su estructura interna, y su comportamiento está caracterizada por un ECC (Execution Control Chart). El ECC para el controlador periódico se muestra en la figura 3, y esá compuesto por tres estados (INIT, RST y REQ) donde se ejecutan dos algoritmos (RESET y REQ) y cuyas transiciones se pasan si se reciben los eventos REQ, RST o INIT y se sale siempre de los estados. En algunos estados (INIT y en REQ), se pueden enviar eventos de salida, como lo son CNF en el caso de REQ y INITO en el caso de INIT. El algoritmo RESET es una puesta a cero de valores (término derivativo, integral y anterior medida), mientras que el algoritmo REQ realiza el cálculo del controlador PID.

El ECC del controlador de Årzen es más complejo, Figura 4. En este caso, se mantienen tanto las partes de inicialización y de reset del controlador pero en cuanto a la ejecución, una vez se recibe el evento REQ, se pasa al estado *Check_point* donde se procede a evaluar el tiempo que ha pasado entre



(a) (b)

Figura 2: Bloques de funciones de los PID periódico (a) y basado en eventos (b).

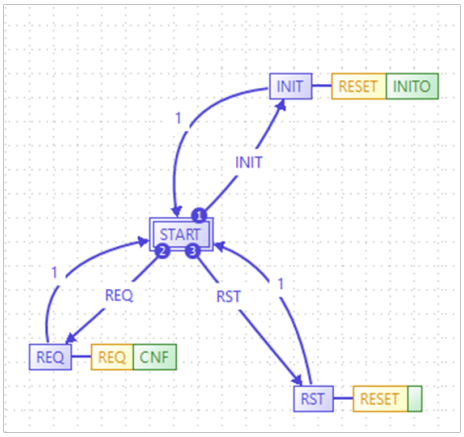


Figura 3: ECC del controlador PID basado en tiempo de muestreo.

la última ejecución y el momento actual mediante el algoritmo *Check_time*. Una vez efectuado este cálculo se comprueban las transiciones, que son las condiciones que impone Årzen para la ejecución de su controlador (la función abs devuelve un entero, por eso es necesario añadir algunas cifras significativas con la parámetro en entrada *E_precision*). En el caso de no tener que ejecutarse simplemente se entra en el estado SENDNOMOD y se envía el evento NO_MOD. En caso contrario, se ejecuta el algoritmo *Req*, donde se realiza el cálculo de la nueva acción de control.

Finalmente, se modela una aplicación donde se incluyen diferentes bloques para la correcta ejecución del algoritmo de control. Se incluyen los bloques XENO_RT_E_CYCLE para llamar regularmente a la cadena de FBs, el bloque BB-Bio_AIn que proporciona la entrada medida, F_RT_CLOCK_NS que proporciona el tiempo del procesador para efectuar cálculos dentro del controlador y un bloque adicional para actualizar la

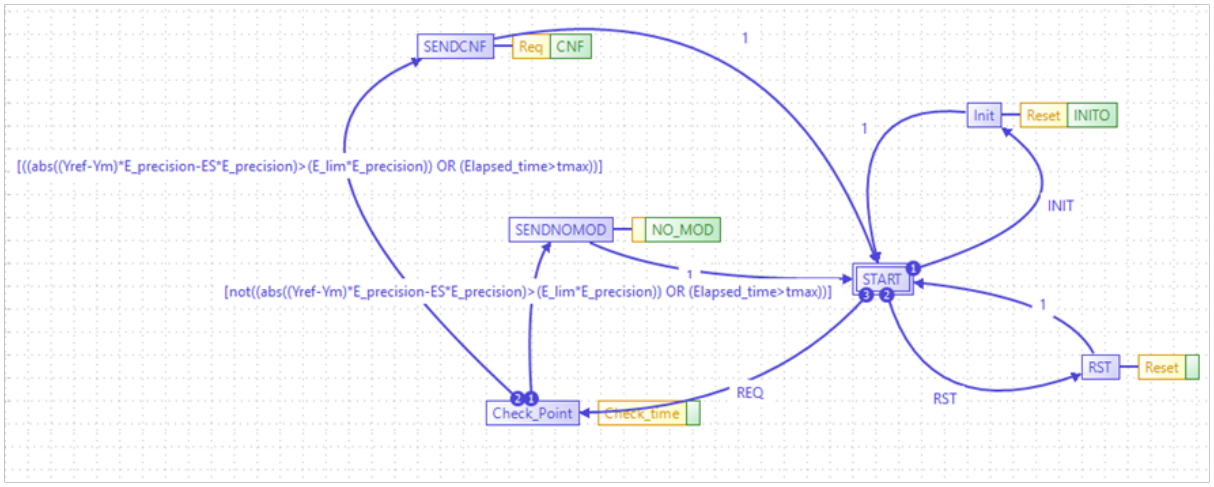


Figura 4: ECC del controlador PID basado en eventos propuesto por Årzen [1].

salida PWM BBBio_PWM, como se muestra en la Figura 5.

La ejecución de la aplicación se desarrolla en dos fases. En una primera fase de inicialización se envía un evento a la entrada INIT del bloque Cycle desde el recurso que contiene esta aplicación. Una vez inicializado este FB se inicia toda la cadena hasta llegar a la salida PWM, siendo este último bloque el que da comienzo a la fase de ejecución con el evento entrante en el START del FB Cycle, iniciando así el envío periódico de eventos y el control del sistema ejecutando o no el controlador.

5. MODELADO DE TIEMPOS DE EJECUCIÓN

Como se ha comentado anteriormente, uno de los objetivos de este trabajo es la evaluación de los controladores basados en eventos desarrollados según el estándar IEC-61499. En dicha evaluación abordaremos tanto el comportamiento del sistema de control en lo relativo a su respuesta, y al coste computacional de este tipo de controladores. La comparación se realizará con un controlador periódico.

En primer lugar, podemos caracterizar el tiempo de realizar Q ejecuciones del controlador periódico como la suma del tiempo que consume cada uno de los FBs que forman su cadena de ejecución por medio de la ecuación (1):

$$t_{PIDtb} = Q(t_{AIN} + t_{PID} + t_{PWM}) \qquad (1)$$

donde t_{AIN} es el tiempo de ejecución del bloque de lectura de entrada analógica (BBBio_AIN), t_{PID} es el tiempo de ejecución del bloque PID y t_{PWM} es tiempo de ejecución del bloque de escritura en la salida analógica (BBBio_PWM).

En el caso del PID basado en eventos, dicha ecuación debe tener en cuenta el porcentaje de generación de eventos, que llamaremos η, o sea, cuando la condición para el cálculo de una nueva acción de control es verdadera y por tanto el algoritmo de control se ejecuta completamente. El tiempo de ejecución es descrito por la ecuación (2) obtenida de la modelización presentada en la figura 5. En este caso t_{PID_e} es el tiempo que consume el bloque PID cuando se tiene que calcular una nueva acción de control, $t_{PID_{ne}}$ es el tiempo consumido por el bloque PID cuando no se tiene que calcular un nuevo valor de acción de control y t_{CLK} es el tiempo que consume el bloque F_RT_CLOCK_NS encargado de la obtención del tiempo del procesador.

$$t_{PIDeb} = Q(\eta(t_{AIN} + t_{PID_e} + t_{PWM} + t_{CLK}) + (1-\eta)(t_{AIN} + t_{PID_{ne}} + t_{CLK})) \qquad (2)$$

Usando las ecuaciones (1) y (2) podemos calcular el porcentaje de tiempo necesario por el controlador basado en eventos respecto del controlador periódico, en función del porcentaje de generación de eventos (η). Dicha relación vendrá dada por el siguiente cociente:

$$\frac{t_{PIDeb}}{t_{PIDtb}} = \frac{Q(\eta(t_{AIN} + t_{PID_e} + t_{PWM} + t_{CLK}))}{t_{AIN} + t_{PID} + t_{PWM}} + \frac{(1-\eta)(t_{AIN} + t_{PID_{ne}} + t_{CLK})}{t_{AIN} + t_{PID} + t_{PWM}} \qquad (3)$$

Si definimos $\gamma = \frac{t_{PIDeb}}{t_{PIDtb}}$ y operamos sobre la ecuación anterior, obtenemos que la relación entre γ y η viene dada por la ecuación (4), que se corresponde a la ecuación de una recta cuyos parámetros dependen de los tiempos de ejecución de cada uno de los bloques implicados en el modelo de la apli-

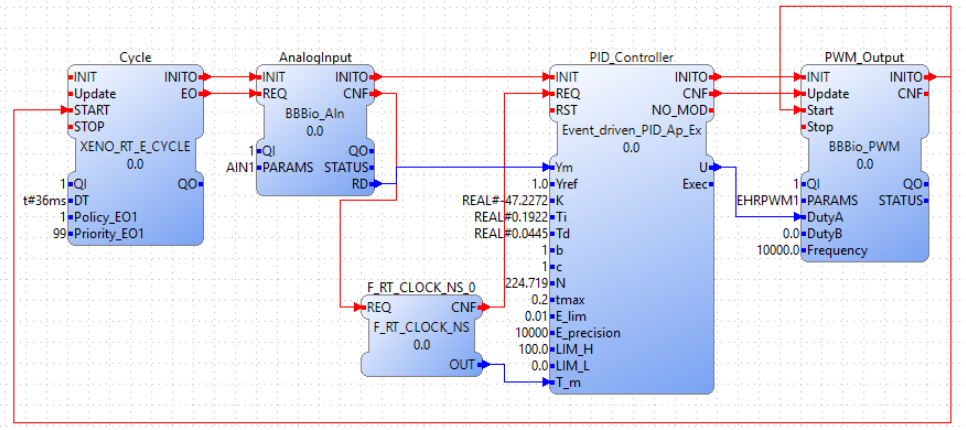


Figura 5: Estructura de la aplicación de control PID basado en eventos según IEC-61499, desarrollada en 4DIAC.

cación.

$$\gamma = \eta \frac{t_{PID_e} - t_{PID_{ne}} + t_{PWM}}{t_{AIN} + t_{PID} + t_{PWM}} + \frac{t_{PID_{ne}} + t_{AIN} + t_{CLK}}{t_{AIN} + t_{PID} + t_{PWM}} \quad (4)$$

A partir de la ecuación anterior podemos llegar a interesantes conclusiones sobre el tiempo consumido por un controlador basado en eventos respecto del controlador periódico en función del porcentaje de ejecución del primero. Por ejemplo, evaluando la ecuación 4 para $\gamma = 1$, obtenemos el porcentaje de generación de eventos η para el cual el controlador basado en eventos tiene el mismo tiempo de ejecución medio que el controlador periódico. Dicho valor vendría dado por la siguiente ecuación:

$$\eta = \frac{t_{PID} - t_{PID_{ne}} + t_{PWM} - t_{CLK}}{t_{PID_e} - t_{PID_{ne}} + t_{PWM}} \quad (5)$$

Si evaluamos la ecuación 4 para $\eta = 0$, obtenemos el tiempo medio consumido por el CBE respecto del controlador periódico para un porcentaje de generación de eventos igual a cero. Dicha condición se corresponde con el funcionamiento del controlador en estado estable, si no se genera evento alguno. La ecuación sería la siguiente:

$$\gamma = \frac{t_{PID_{ne}} + t_{AIN} + t_{CLK}}{t_{AIN} + t_{PID} + t_{PWM}} \quad (6)$$

Finalmente, si evaluamos la ecuación (4) para $\eta = 1$, se obtiene el tiempo medio consumido por el controlador basado en eventos respecto del controlador periódico para un 100 % de generación de eventos. Dicho comportamiento se corresponde con el comportamiento del controlador en estado transitorio, cuando se produce un cambio en la referencia o aparece una perturbación de valor significativo.

Los resultados obtenidos con estas ecuaciones para la plataforma de experimentación usada en este estudio serán presentadas en la siguiente sección.

6. RESULTADOS EXPERIMENTALES

6.1. Plataforma de experimentación

La plataforma experimental para la evaluación del comportamiento de los controladores basados en eventos está formada por una tarjeta BeagleBone Black (BBB). Dicha tarjeta cuenta con varias entradas analógicas y salidas PWM que la hacen idónea para realizar nuestros experimentos, a diferencia de otras alternativas que también fueron valoradas como la tarjeta Raspberry Pi.

Con el objetivo de tener una plataforma experimental lo más simple y versátil posible, se optó por definir los sistemas a controlar mediante circuitos electrónicos cuya dinámica viene determinada por los componentes pasivos que lo forman. En concreto se trata de un filtro paso-bajo de segundo orden con una estructura de Rauch, cuyo esquema se muestra en la Figura 6, con función de transferencia definida por sus componentes de acuerdo con la ecuación (7).

$$H(s) = -\frac{\frac{1}{R^2 C_1 C_2}}{s^2 + \frac{3}{RC_2}s + \frac{1}{R^2 C_1 C_2}} \quad (7)$$

Cuatro de estos circuitos fueron incluidos en una tarjeta electrónica que se puede acoplar directamente sobre la BBB en forma de capa, como se muestra en la Figura 7. Los componentes que definen la dinámica de los circuitos se pueden cambiar fácilmente, por lo que resulta muy sencillo tener sistemas con diferentes dinámicas.

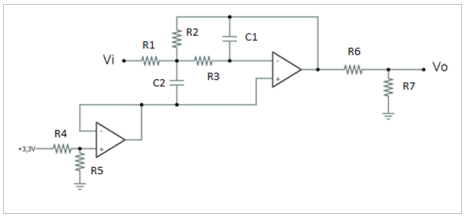


Figura 6: Circuito electrónico que emula un sistema a controlar.

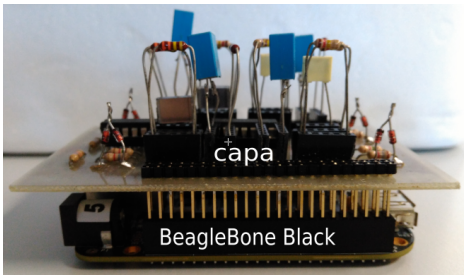


Figura 7: Conjunto BeagleBone Black y capa con circuitos a controlar.

Para programar la tarjeta BBB y hacer los experimentos, la conectamos a un ordenador y mediante una conexión SSH accedemos al directorio donde se encuentra el runtime FORTE (con los bloques que queremos probar compilados) y lo ejecutamos. Acto seguido, descargamos la aplicación desde el ordenador a la BBB con el programa 4DIAC y comienza su ejecución automáticamente.

6.1.1. Sistema de tiempo real: Xenomai

Las tarjetas BBB traen instalado por defecto un sistema operativo Linux que no permite ejecutar aplicaciones en tiempo real. Esto supone un problema ya que la ejecución de cada nuevo FB se realiza cada 5 ms independientemente de los tiempos de ejecución de cada uno. Para mejorar las prestaciones de la BBB se ha instalado en ella Xenomai. Esto nos proporciona una gestión de la aplicación en tiempo real y permite reducir de forma significativa los tiempos de respuesta de la tarjeta.

6.2. Resultados y discusión

6.2.1. Tiempos de ejecución

Para poder aplicar la ecuación (4) a nuestras aplicaciones de control periódico y basado en eventos al ejecutarse en la BBB se han realizado mediciones de los tiempos de ejecución medio de cada uno de los FB que intervienen en ellas. Para ello hemos medido con un osciloscopio el tiempo entre la activación de una salida digital de la BBB que tiene lugar al inicio de la ejecución del FB y la desactivación de la misma salida digital que tiene lugar al final de la ejecución del FB en estudio. Con el objetivo de corregir la medición realizada de esta forma con el tiempo requerido de activación y desactivación de una salida digital, se hizo un test preliminar en que hemos activado una salida digital para acto seguido desactivarla. El tiempo obtenido ha sido de entorno a los 4 μs. Este valor se restó a los tiempos medidos para cada uno de los FB. Los resultados arrojados por el estudio se muestran en la Tabla 1.

Tabla 1: Resultados experimentales de tiempos de ejecución medios.

Tiempos	Valor (μs)
t_{AIN}	8
t_{PWM}	12.7
t_{PID}	8.5
t_{PID_e}	10.5
$t_{PID_{ne}}$	5.5
t_{CLK}	2

Usando los datos de la tabla 1 en la ecuación 4, obtenemos la recta mostrada en la figura 8, de la cual podemos extraer como resultados importantes los siguientes:

- El tiempo de ejecución medio del controlador basado en eventos es igual al requerido por el controlador periódico si el porcentaje de generación de eventos del primero es del 77.12 %.
- Si la tasa de generación de eventos es del 100 % entonces el tiempo de ejecución medio del controlador basado en eventos sobrepasa al tiempo de ejecución del controlador periódico en un 13.8 % de su valor.
- Por último, si no se generan eventos, entonces el tiempo de ejecución medio del controlador basado en eventos es el 53.45 % del tiempo de ejecución del controlador periódico.

Respecto a este último punto, debemos tener en cuenta que para el controlador propuesto por Årzén no existe una tasa de generación de eventos igual a cero ya que el algoritmo fija un tiempo máximo (`hmax`) que si es sobrepasado se genera un evento. Si se considera que el tiempo de ejecución del algoritmo es hnom, entonces la tasa de generación de eventos para una secuencia de eventos por superación de tiempo máximo será hnom/hmax. Esta sería la tasa de generación mínima que puede alcanzar del algoritmo de Årzén.

6.2.2. Comportamiento del sistema

Para el estudio del comportamiento del sistema de control se consideró un circuito como el mos-

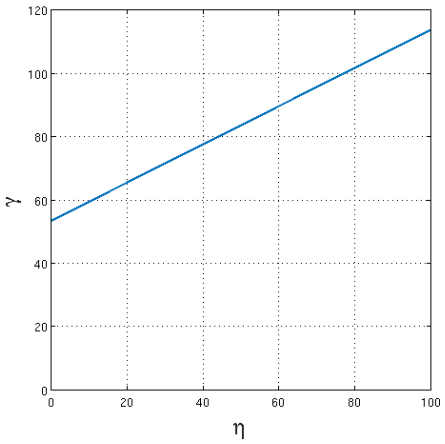


Figura 8: Relación entre η y γ.

trado en la figura 6 con función de transferencia dada por la ecuación 8, la cual se obtiene con los siguientes valores de los componentes: C1=270nF, C2=1μF, R1=R2=R3=330$k\Omega$, R4=R5=1$k\Omega$, R6=1.5$k\Omega$ y R7=1.8$k\Omega$.

$$G(s) = \frac{0{,}612}{s^2 + 9s + 34} \tag{8}$$

El ajuste de los parámetros del controlador PID se realizó mediante el método AMIGO presentado en [2]. Para el diseño se ha usado un valor máximo de la función de sensibilidad Ms=1.4 y un coeficiente de filtro de la derivada N=10. Los parámetros obtenidos son los siguientes: Kc=47.2, Ti=0.19, Td=0.044. Estos parámetros fueron usados tanto para el controlador periódico como para el basado en eventos. El periodo de muestreo en ambos casos se fijó en hnom=36ms y los valores de ponderación de la referencia y de la derivada de la referencia se consideraron b=1 y c=1. Además los valores de saturación mínimo y máximo de la acción de control: LIM_H=5 y LIM_L=0. Para el controlador basado en eventos se consideró umbral de cruce E_lim=0.01 y un tiempo máximo de ejecución sin eventos hmax=5hnom.

La Figura 9 muestra las respuestas obtenidas con el controlador periódico y el CBE propuesto por Årzén ante tres cambiso en la referencia en $t = 15$, 20 y 25 segundos. Se puede observar que la respuesta del PID periódico es similar en los tres cambios de referencias, mientras que el CBE tiene diferencias significativas en el tercer cambio respecto a los dos primeros. Esto es consecuencia de que en el CBE la respuesta está muy influenciada por la casuística de generación de los eventos. Se puede notar, además, que la acción de control del CBE tiene cambios más bruscos que la del controlador periódico. Esto se debe al efecto de la acción integral, según se indicó por Durand y sus

Tabla 2: Indices IAE y TV de la respuesta del sistema real con ambos controladores.

Indice	Experimento	Periódico	Arzén
IAE	15sec	0.103	0.0948
	20sec	0.1035	0.0969
	25sec	0.1031	0.0792
TV	15sec	0.6378	0.6431
	20sec	0.6106	0.6747
	25sec	0.6264	0.5323

colaboradores en [5]. Con el objetivo de cuantificar las diferencias entre ambas respuestas se ha calculado el índice IAE de la salida y el TV de la acción de control para cada controlador. Los valores obtenidos se muestran en la Tabla 2, donde se puede apreciar que no existen diferencias significativas entre ambos controladores. Los resultados confirman la variabilidad en las respuestas del CBE respecto del controlador periódico: para el tercer cambio de referencia los valores obtenidos para el CBE son diferentes respectos a los resultado para $t = 15$ y $t = 20$ segundos.

En la zona de estado estable se puede apreciar como el controlador basado en eventos se ejecuta por tiempo máximo cada 5hnom, lo cual quiere decir una tasa de generación de eventos η=hnom/hmax=1/5=20 %. Para este valor de η se obtiene de la gráfica de la Figura 8 que tiempo medio de ejecución medio del controlador de Årzén es el 66 % del tiempo medio de ejecución del controlador periódico. Por el contrario, cuando se produce un cambio en la referencia el controlador basado en eventos se ejecuta todos los instantes de muestreo, o sea que la tasa de generación de eventos es $\eta = 100\,\%$ y en ese caso el controlador de Årzén requiere para su ejecución el tiempo del controlador periódico incrementado en un 13.8 % de su valor, como indica el valor de γ en la Figura 4. Durante la respuesta transitoria la tasa de generación de evento η varía, y en consecuencia lo hace γ.

7. CONCLUSIONES

En este artículo se ha presentado la implementación de controladores basados en eventos según el estándar IEC-61499 para la programación de sistema de control distribuidos. Además se ha realizado un estudio donde se ha tenido en cuenta tanto las aspectos relativos al coste computacional de los algoritmos como la respuesta del sistema de control.

En cuanto al coste computacional se ha obtenido un modelo que permite comparar de una forma sistemática los controladores basados en eventos con los periódicos. Dicho modelo relaciona la tasa de generación de eventos de un controlador basado en

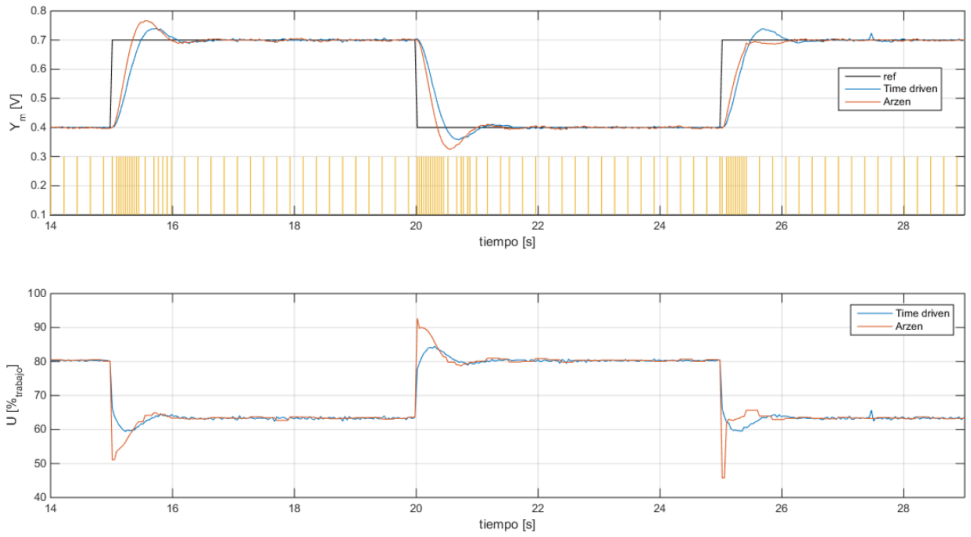


Figura 9: Respuesta del sistema (superior)y acción de control (inferior) a cambios en referencia controlado con un PID periódico y el PID basado en eventos propuesto por Årzén. Las líneas verticales amarillas en la gráfica superior representan los eventos generados por el controlador de Årzén.

eventos con el porcentaje de tiempo de ejecución de un controlador períodico usado por el algoritmo basado en tiempo. El modelo permite conocer el ahorro en tiempo de ejecución que se obtiene para una tasa de generación de eventos baja, o el sobre coste en tiempo de ejecución para tasa de generación de eventos altas, tomando como referencia el tiempo de ejecución del controlador periódico.

Se ha demostrado de forma experimental la viabilidad del estándar IEC-6499 para la implementación de algoritmos de control basados en eventos. Para ello se ha desarrollado una plataforma que permite realizar experimentos considerando sistemas con diferentes características de respuesta dinámica.

Agradecimientos

Este trabajo ha sido financiado mediante fondos de los proyectos TEC2015-69155-R del MICINN y P1·1B2015-42 de la Univesitat Jaume I.

Referencias

[1] Karl-Erik Arzén. A simple event-based pid controller. In *Proc. 14th IFAC World Congress*, volume 18, pages 423–428, 1999.

[2] K.J. Åström and T. Hägglund. Revisiting the Ziegler–Nichols step response method for PID control. *Journal of Process Control*, 14(6):635–650, sep 2004.

[3] Sebastián Dormido, J Sánchez, and Ernesto Kofman. Muestreo, control y comunicación basados en eventos. *Revista Iberoamericana de Automática e Informática Industrial RIAI*, 5(1):5–26, 2008.

[4] Sylvain Durand and Nicolas Marchand. An event-based pid controller with low computational cost. In *8th International Conference on Sampling Theory and Applications (SampTA'09)*, pages Special–session, 2009.

[5] Sylvain Durand and Nicolas Marchand. Further results on event-based pid controller. In *Control Conference (ECC), 2009 European*, pages 1979–1984. IEEE, 2009.

[6] Karl-Heinz John and Michael Tiegelkamp. *IEC 61131-3: programming industrial automation systems: concepts and programming languages, requirements for programming systems, decision-making aids*. Springer Science & Business Media, 2010.

[7] Volodymyr Vasyutynskyy and Klaus Kabitzsch. Time constraints in pid controls with send-on-delta. *IFAC Proceedings Volumes*, 42(3):48–55, 2009.

[8] Volodymyr Vasyutynskyy and Klaus Kabitzsch. A comparative study of pid control algorithms adapted to send-on-delta sampling. In *Industrial Electronics (ISIE), 2010 IEEE International Symposium on*, pages 3373–3379. IEEE, 2010.

[9] Alois Zoitl and Robert Lewis. *Modelling control systems using IEC 61499*, volume 95. IET, 2014.